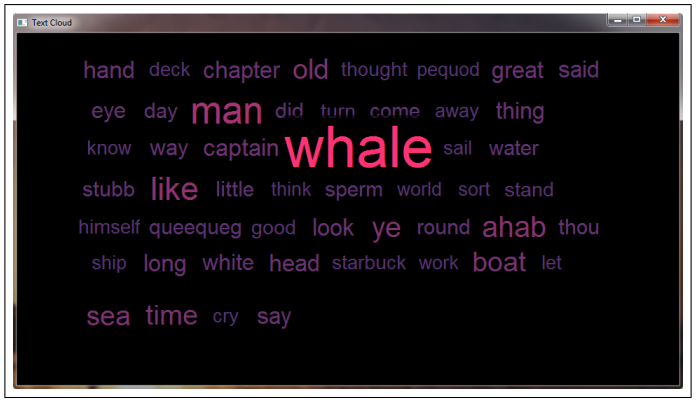
CSCI 203 Place-out Project
final.pdf

For the final part of CSCI 203 final project, I implemented a web crawler, drawing inspiration from what I learned during an online course CS101: Building a Search Engine. The web crawler collects the text from all web pages with links present on the user-entered URL. The process continues until DEPTH number of "hops" from the original page are made. For example, only the first web page is crawled if DEPTH is set to 0. However, even the pages linked from that page are searched if DEPTH is increased to 1. A simple if statement is used to ensure that all pages are visited exactly once.

I revamped the program simpleTextCloudDisplay.py to create a more organized text cloud. The modified program named as complexTextCloudDisplay.py has the following functions:

- windowSetup()
  Sets up the VPython window "scene"
- extreme(freqList, max_min)
  Returns position of maximum frequency if max_min is True, position of minimum frequency otherwise
- findHeight(freqList, pos)
  Returns height (font size) of a word on position pos
- findWidth(freqList, pos)
  Returns approximate "screen width of word" on position pos
- displayCloud(freqList)
  Invokes windowSetup() and displays a Text Cloud

Furthermore, I polished the stemContent() by including a Dict named as exceptions. The stemming process is bypassed for any word which is one of the keys in exceptions, and its corresponding value is then treated as the stemmed word. Following is a sample program output with MAX_WORDS and DEPTH set to 50 and 2 respectively:



```
Please enter a URL
   http://www.eg.bucknell.edu/~csci203/common-files/sampleText/moby-dick.html
[('whale', 1392), ('man', 762), ('like', 588), ('ahab', 498), ('ye', 486),
   ('sea', 466), ('old', 446), ('time', 436), ('boat', 426), ('captain',
   337), ('head', 333), ('hand', 329), ('look', 326), ('long', 324),
   ('great', 324), ('chapter', 315), ('thing', 312), ('say', 311), ('said',
   293), ('way', 278), ('white', 276), ('eye', 268), ('thou', 264),
   ('stubb', 257), ('round', 256), ('did', 251), ('little', 249), ('day',
   248), ('sperm', 241), ('queequeg', 239), ('water', 231), ('come', 226),
   ('turn', 213), ('stand', 203), ('good', 203), ('himself', 200), ('know',
   194), ('starbuck', 189), ('sail', 186), ('thought', 183), ('let', 183),
   ('deck', 181), ('world', 177), ('away', 176), ('pequod', 175), ('work',
   172), ('ship', 172), ('sort', 171), ('think', 169), ('cry', 166)]
```

---

1. Following are three functions which can be reused in similar programs:

   - `ModifiedStemmer.stem()`
     Most search engine crawl web pages for specific information. This function could be used in a web search engine algorithm to remove suffixes from query words. Such a function would even list the otherwise neglected results.
   - `complexTextCloudDisplay.displayCloud()`
     This function could be put to use whenever a visual display is needed. Facebook and Twitter could implement such a method to display a text cloud of all trending topics wherein each topic link to a corresponding news feed.
   - `project_part1.getContent()`
     If a website does not provide an Application Program Interface (API), this function could be used for basic web scraping, which means to extract useful information from web pages.

   I enjoyed implementing a couple of program features. First of all, the `ModifiedStemmer Class` definition uses slick logic to minimize the lines of code while accounting for most major suffixes. Even though `stemMajor()` function is a bit hard to read, the overall efficiency of my stemmer is on a higher side.

   Throughout the program, I try to optimize space complexity. In `cleanContent()`, `stopwordsIndices` maintains the indices of stopwords in `wordList`. Instead of appending to `stopwordsIndices` though, a stopword index is inserted at its beginning. This avoids `IndexError` as the program deletes stopwords from right to left. Additionally, I make local copies of `MAX_WORDS` and `DEPTH` so that the program does not encounter errors due to `global` variables.

2. After testing my program files multiple times, I feel that a considerable portion of this project should be working fine. Though there could be several "doom scenarios" that could result in anomalous behavior. If the web crawler encounters `www.bucknell.edu` or any similar generic `URL` with massive number of links, the program can take a lot

of time to gather content and may even cause the program to crash due to excessive memory use.

Also, randomization of `freqList` in `displayCloud()` could result in undesirable spacing in the following case. If a word is among the more frequent words in `freqList` and is to be put at the beginning of a line, the resulting window scene would have unwanted space before that particular line.

3. Given more time, following features could be incorporated to my project:

   - Improve display by using a `Tkinter Canvas` instead of `Labels`. While searching for documentation on `scene` and `Label`, I discovered that `Canvas` would be simpler to display a Text Cloud and would subsequently reduce hard-coding coordinate values.
   - Account for more suffixes in `ModifiedStemmer Class`. In the original paper by Martin Porter, there are five levels of stemming, last two of which cover less prominent suffixes. Efficiency of my stemmer could be further increased if these suffixes are considered too.
   - Make the program more user-friendly. This can be done by outputting helpful messages throughout program run. For instance, if a user enters an invalid `URL` at first, a message asking if (s)he would like to continue with the program and enter another `URL` could go a long way.

References & Useful Links

[1] Martin F. Porter. An algorithm for suffix stripping. *Program* 14.3: pp. 130–137, 1980.

[2] List of English Stop Words. `http://xpo6.com/list-of-english-stop-words/`.

[3] HSL Colors. `http://www.css3.info/preview/hsl/`.

[4] `https://docs.python.org/3/tutorial/datastructures.html`.

[5] `https://docs.python.org/2/library/string.html`.

[6] `http://www.vpython.org/webdoc/visual/display.html`.

[7] Modifying global variable with same name as local variable. `http://stackoverflow.com/questions/10235973/modifying-global-variable-with-same-name-as-local-variable`.

[8] Best way to randomize a list of strings in Python. `http://stackoverflow.com/questions/1022141/best-way-to-randomize-a-list-of-strings-in-python`.